Efficient Runahead Execution Processors

Publication No. _____

Onur Mutlu, Ph.D.
The University of Texas at Austin, 2006

Supervisor: Yale N. Patt

High-performance processors tolerate latency using out-of-order execution. Unfortunately, today's processors are facing memory latencies in the order of hundreds of cycles. To tolerate such long latencies, out-of-order execution requires an instruction window that is unreasonably large, in terms of design complexity, hardware cost, and power consumption. Therefore, current processors spend most of their execution time stalling and waiting for long-latency cache misses to return from main memory. And, the problem is getting worse because memory latencies are increasing in terms of processor cycles.

The runahead execution paradigm improves the memory latency tolerance of an out-of-order execution processor by performing potentially useful execution while a long-latency cache miss is in progress. Runahead execution unblocks the instruction window blocked by a long-latency cache miss allowing the processor to execute far ahead in the program path. This results in other long-latency cache misses to be discovered and their data to be prefetched into caches long before it is needed.

This dissertation presents the runahead execution paradigm and its implementation on an out-of-order execution processor that employs state-of-the-art hardware prefetching

techniques. It is shown that runahead execution on a 128-entry instruction window achieves the performance of a processor with three times the instruction window size for a current, 500-cycle memory latency. For a near-future 1000-cycle memory latency, it is shown that runahead execution on a 128-entry window achieves the performance of a conventional processor with eight times the instruction window size, without requiring a significant increase in hardware cost and complexity.

This dissertation also examines and provides solutions to two major limitations of runahead execution: its energy inefficiency and its inability to parallelize dependent cache misses. Simple and effective techniques are proposed to increase the efficiency of runahead execution by reducing the extra instructions executed without affecting the performance improvement. An efficient runahead execution processor employing these techniques executes only 6.2% more instructions than a conventional out-of-order execution processor but achieves 22.1% higher Instructions Per Cycle (IPC) performance.

Finally, this dissertation proposes a new technique, called address-value delta (AVD) prediction, that predicts the values of pointer load instructions encountered in runahead execution in order to enable the parallelization of dependent cache misses using runahead execution. It is shown that a simple 16-entry AVD predictor improves the performance of a baseline runahead execution processor by 14.3% on a set of pointer-intensive applications, while it also reduces the executed instructions by 15.5%. An analysis of the high-level programming constructs that result in AVD-predictable load instructions is provided. Based on this analysis, hardware and software optimizations are proposed to increase the benefits of AVD prediction.